Technikerschule HF Zürich – Software Engineering 1

»

«

Projektarbeit:

YourSwissQR



**Fabrizio Piacente**

**Nicolas Hässig**

**Markus Kaufmann**

**30.09.2020**

**TS TSI 1909 A 03 SoE**

Inhalt

[Vorwort 4](#_Toc56976946)

[1. Aufgabe 1: Projektumgebung definieren und dokumentieren 5](#_Toc56976947)

[1.1 Richtlinien und Schreibweisen 5](#_Toc56976948)

[1.2 Aufbau der Ordnerstruktur 6](#_Toc56976949)

[1.3 Eingesetzte Entwicklungssoftware 7](#_Toc56976950)

[1.4 Checklisten und Installationsrichtlinien der Entwicklungssoftware 7](#_Toc56976951)

[1.5 Organisation der Arbeitsgruppe 8](#_Toc56976952)

[1.5.1 Kommunikation im Team 8](#_Toc56976953)

[1.5.2 Austausch der Arbeitsergebnisse 8](#_Toc56976954)

[1.5.3 Aufgabenverteilung 8](#_Toc56976955)

[1.5.4 Projektplanung 8](#_Toc56976956)

[1.6 Projektentscheidungen 9](#_Toc56976957)

[1.7 Glossar wichtiger Begriffe 9](#_Toc56976958)

[2. Aufgabe 2: Projektauftrag erstellen und präsentieren 10](#_Toc56976959)

[2.1 Unternehmen 10](#_Toc56976960)

[2.2 Projekt 11](#_Toc56976961)

[2.2.3 Domäne 11](#_Toc56976962)

[2.2.4 Problemstellung 11](#_Toc56976963)

[2.2.5 Projektbeteiligte 11](#_Toc56976964)

[2.2.6 Projektplanung: 1. Iteration 12](#_Toc56976965)

[2.3 Anforderungsanalyse 13](#_Toc56976966)

[2.3.1 Vision 13](#_Toc56976967)

[2.3.2 Name 13](#_Toc56976968)

[2.3.3 Stakeholder 13](#_Toc56976969)

[2.3.4 Funktionale Anforderungen 14](#_Toc56976970)

[2.3.5 Nicht-funktionale Anforderungen 16](#_Toc56976971)

[2.3.6 Benutzer 16](#_Toc56976972)

[2.3.7 Umsysteme 17](#_Toc56976973)

[2.3.8 Systemkontextdiagramm 17](#_Toc56976974)

[3. Aufgabe 3: Objektorientierte Analyse (OOA) 18](#_Toc56976975)

[3.1 Fachklassenmodell 18](#_Toc56976976)

[3.2 Zustandsmodell 19](#_Toc56976977)

[3.3 Verantwortlichkeit Fachklassen 19](#_Toc56976978)

[4. Aufgabe 4: Designmodell erstellen (OOD) 20](#_Toc56976979)

[4.1 Designmodell für die Persistence Schicht 20](#_Toc56976980)

[4.2 Designmodell für die Business-Schicht 20](#_Toc56976981)

[4.3 Integriertes Designmodell 21](#_Toc56976982)

[4.4 Sequenz Diagramm 24](#_Toc56976983)

[4.5 Komponenten Diagramm 25](#_Toc56976984)

[5. Glossar 26](#_Toc56976985)

[5.1 Begriffserklärung 26](#_Toc56976986)

[5.2 Klassenübersicht 27](#_Toc56976987)

[6. Tabellen und Abbildungsverzeichnis 28](#_Toc56976988)

[6.1 Abbildungsverzeichnis 28](#_Toc56976989)

[6.2 Tabellenverzeichnis: 28](#_Toc56976990)

[7. Quellenverzeichnis 28](#_Toc56976991)

# Vorwort

Im Rahmen des Modules «Software Engineering 1» wird gruppenweise ein komplettes Software Projekt geplant und umgesetzt. Neben der eigentlichen Implementation des Projektes sind sämtliche Planungs- und Entwurfsschritte, sowie die planerische Umsetzung zu dokumentieren.

Dieses Dokument enthält die Umsetzung der schriftlichen Arbeitsaufträge. Dabei ist jede Aufgabe als eigenes Hauptkapitel aufgeführt.

Als Projekt Thema wurde das Erstellen von SwissQR-Codes aus Debitorenrechnungsinformationen gewählt. Der SwissQR-Code, welcher am 30.Juni 2020 eingeführt wurde, wird bis im Jahr 2022 die bestehenden «Orangen» ESR Einzahlungsscheine ablösen, weshalb das gewählte Thema eine tatsächliche Aktualität bedient.

# Aufgabe 1: Projektumgebung definieren und dokumentieren

Die Aufgabe eins hat die Definition und Verwaltung des gesamten Projektauftrags zum Ziel. Diverse Unterpunkte wie zum Beispiel die Beschreibung der Styleguide sind deshalb in den folgenden Unterkapiteln enthalten. Das zu führende Glossar ist als eigenes Kapitel am Ende des Dokuments enthalten.

## Richtlinien und Schreibweisen

Um zu gewährleisten das der gesamte Programmcode einheitlich geschrieben wird, wird zu Projektbeginn eine Styleguide festgelegt, die zwingend einzuhalten ist. Zudem werden auch Richtlinien für die Benamsung der Projektstruktur (Ordner und Files) definiert.

Folgende Richtlinien sind festgelegt:

|  |  |
| --- | --- |
| Ordnernamen | Für Ordnernamen sind sprechende englische Bezeichnungen zu vergeben und in Camel-Case zu schreiben (Bsp. «ProjectDocumentation») |
| Dateinamen | Als Dateinamen sind sprechende englische Bezeichnungen zu vergeben und in Camel-Case zu schreiben (Bsp. «TaskOnePresentation.pptx») |
| Codingstyle (Java) | Als Java Coding Styleguide wird die Google Java Style Guide verwendet. Wir behalten es uns zu diesem Zeitpunkt offen, diese, um eigene Richtlinien zu erweitern. Anbei der Link zu der Styleguide: <https://google.github.io/styleguide/javaguide.html>. |

Tabelle Richtlinien und Schreibweisen

## Aufbau der Ordnerstruktur

In diesem Unterkapitel wird der Aufbau des Projektes beschrieben, dabei wird darauf eingegangen, wo welche Komponenten des Projekts zu finden sind.

Das Projekt wurde mit folgender Ordnerstruktur initiiert:

* Root
  + Documentation
    - Documents
    - Presentations
    - ProjectOrganisation
  + Project

Im Ordner «Documentation» werden sämtliche Dateien zum/um den Projektaufbau und Dokumentation abgelegt. Im Unterordner «Documents» sind zudem externe Dokumente, wie zum Beispiel Richtlinien zum SwissQR Code enthalten. Im Presentations Ordner werden die zu haltenden Präsentationen gesammelt und der Ordner ProjectOrganisation ist als Sammelbecken für planerische Aspekte des Projekts gedacht.

Der Project Ordner wird die komplette Java Solution enthalten und ist momentan bis auf ein Test Java Projekt leer.

## Eingesetzte Entwicklungssoftware

Sämtliche Software, welche für die Bearbeitung des Projektes verwendet wird, ist an dieser Stelle aufgelistet und kurz beschrieben. Die Anleitung, wie die einzelnen Softwares zu installieren und konfigurieren sind, befindet sich unter dem Kapitel 1.4.

**Entwicklungssoftware:**

|  |  |
| --- | --- |
| Eclipse IDE for Enterprise Java Developers (Version: 2020-09 (4.17.0)) | Entwicklungs IDE für die Bearbeitung und Kompilierung des JAVA Codes  <https://www.eclipse.org/eclipse/news/4.17/> |
| GitHub mit Smartgit (20.1.5) | Git Source Save System für die Verwaltung des Quellcodes und der Dokumentationsdateien  <https://www.syntevo.com/smartgit/download/> |

Tabelle Eingesetzte Entwicklungssoftware

**Weitere verwendete Software:**

|  |  |
| --- | --- |
| Notepad++ (v.7.8.8) | Einfacher Texteditor zur Bearbeitung von Textdateien u.ä.  <https://notepad-plus-plus.org/downloads/> |
| Microsoft Office 365 Palette   * Word * PowerPoint * Teams | Für das Erstellen der Dokumentation, Kommunikation im Team und Präsentation eingesetzt.  <https://www.office.com/> |
| Visual Paradigm | UML Software, welche für die Analyse- und Designentwürfe verwendet wird. |

Tabelle Weitere verwendete Software

## Checklisten und Installationsrichtlinien der Entwicklungssoftware

In diesem Unterkapitel wird beschrieben, wie die unter 1.3 genannten Entwicklungssoftware zu nutzen, respektive zu installieren ist.

|  |  |
| --- | --- |
| Eclipse IDE for Enterprise Java Developers (Version: 2020-09 (4.17.0)) | Um Eclipse zu installieren, kann diese Anleitung verwendet werden:  <https://www.eclipse.org/downloads/packages/installer>  Es ist zudem zu beachten, dass die  Java Version 14 (Java Plattform, Standard Edition (JDK)) <https://jdk.java.net/15/>  Installiert und als Pfadvariable gesetzt ist. |

Tabelle Checkliste und Installationsanleitung Eclipse

## Organisation der Arbeitsgruppe

Die gesamte Projektplanung inklusive der Beschreibung der Aufgabenverteilung und der genutzten Hilfsmittel zur Projektkoordination werden an dieser Stelle beschrieben.

### 1.5.1 Kommunikation im Team

Neben Treffen, bei denen alle Projektbeteiligten anwesend sind, werden folgende Hilfsmittel zur Kommunikation im Team genutzt.

* WhatsApp
  + Chat, um Treffen zu vereinbaren und einfache Fragen zu klären
* Microsoft Temas
  + Video Chat für den Austausch bei komplexeren Fragen und das Lösen von Problemstellungen, die gemeinsame Absprachen erfordern.

### 1.5.2 Austausch der Arbeitsergebnisse

Das gesamte Projekt inklusive Dokumentation ist in einem GitHub Repository (<https://github.com/nhaessig/swiss-qr-code>) gesichert, auf welches alle Teammitglieder Zugriff haben. So ist sichergestellt, dass die Arbeitsergebnisse sowohl zentral gesichert als auch jederzeit zur Verfügung stehen. Eigene Änderungen können problemlos commited und gepusht werden, so dass sie in kürzester Zeit dem gesamten Team zur Verfügung stehen.

### 1.5.3 Aufgabenverteilung

Die Aufgabenverteilung innerhalb des Temas wird an dieser Stelle beschrieben, wobei diese sich auf einzelne Tasks aus der Projektplanung bezieht.

### 1.5.4 Projektplanung

In diesem Kapitel wird die Projektplanung aufgeführt, wobei die einzelnen Arbeitsblöcke thematisch und in einer Zeitlinie mit der geplanten und der tatsächlich benötigten Bearbeitungszeit aufgeführt sind.

## Projektentscheidungen

Im Unterkapitel Projektentscheidungen werden richtungsweisende Beschlussfassungen aufgeführt. Dabei wird neben der tatsächlichen Entscheidung jeweils auch das Datum der Entscheidung aufgeführt.

|  |  |  |
| --- | --- | --- |
| Entscheidung/Tätigkeit | Beschreibung | Datum |
| Formen der Projektgruppe | Im ersten Block des Moduls Software Engineering 1 wurde die Projektgruppe, bestehend aus Nicolas Hässig, Markus Kaufmann und Fabrizio Piacente, gebildet. | 22.09.2020 |
| Wahl des Projektthemas (Your SwissQR) | Die Mitglieder der Projektteams haben sich gemeinsam auf die Bearbeitung des folgenden Projektthemas geeinigt: Erstellen SwissQR für Debitorenrechnungen | 28.09.2020 |

Tabelle Projektentscheidungen

## Glossar wichtiger Begriffe

Das Glossar ist als eigenes Kapitel zum Schluss in diesem Dokument enthalten und führt/erklärt die wichtigsten Begriffe dieser Projektarbeit

# Aufgabe 2: Projektauftrag erstellen und präsentieren

Dieses Kapitel enthält die Umsetzung des zweiten Arbeitsauftrags, welcher die Beschreibung des Projektkontext und der Anforderungen an das Softwaresystem zum Thema hat. Die Präsentation des hier beschriebenen Projektauftrags wird als eigene PowerPoint Präsentation erstellt und ist in diesem Dokument nicht enthalten.

## Unternehmen

Im nachfolgenden Kapitel wird das Unternehmen beschrieben, für welches das Software-System entwickelt und umgesetzt wird. Es ist zu beachten, dass die hier beschriebene Firma nicht existiert, weshalb Namen und Kontaktdaten ausgedacht sind.

Die Koch AG wurde im Jahr 2001 als Einzelunternehmen gegründet und bietet ausgewählten Kunden diverse IT-Dienstleistungen.

Spezialisiert ist die Koch AG auf den Bereich Software Entwicklung für Standard und Individual Business Lösungen für KMU’s und kantonale Behörden. Insbesondere im Finanzbereich ist die Koch AG ein starker Partner, bekannte Produkte wie z.B. PayBuddy sind schweizweit im Einsatz.

Rund 32 Mitarbeiter sind bei der Koch AG beschäftigt und entwickeln die Produkte täglich weiter.

Neben der Entwicklung sorgen auch die Projektleiter der Koch AG dafür, dass die Produkte auf die jeweiligen Bedürfnisse zugeschnitten werden.

Die wichtigsten Eckdaten der Koch AG in Tabellenform aufgeführt

|  |  |
| --- | --- |
| Name | Koch AG |
| Geschäftsform | Aktiengesellschaft |
| Geschäftsfeld | Software Entwicklung, Business Lösungen für KMUS/Kantonale Behörden |
| Abteilungen | HR, Sales, Entwicklung, Testing, Projektleitung, Finance |
| Mitarbeiter | 32 |
| Kunden | Kantonale Behörden, Treuhandbereich, KMUS |
| Bekannte Produkte/Software Lösungen | PayBuddy  EasyKredi  PersonaFind |

Tabelle Koch AG Übersicht

## Projekt

Unter dem Unterkapitel «Projekt» wird der geplante Projektauftrag näher beschrieben, es wird näher auf die Projektbeteiligten und die eigentliche Problemstellung eingegangen.

### 2.2.3 Domäne

Das Projekt ist im Rechnungswesen anzusiedeln, Ziel ist die Schaffung einer Debitorenrechnungsverwaltung, welche unter anderem auch die eigentliche Rechnungsstellung an Kunden beinhaltet.

### 2.2.4 Problemstellung

Seit dem 30. Juni 2020 werden in der Schweiz schrittweise die verschiedenen bisher benutzten Einzahlungsscheine durch den neuen SwissQR-Einzahlungsschein abgelöst.

Bis zum Jahr 2020 müssen sämtliche inländischen Rechnung in diesem Format erstellt werden. Rechnungssteller müssen deshalb in der Lage sein, Ihre Rechnungen als SwissQR Rechnung zu erzeugen. Mit unserer Software ist dies einfach um zu setzen und ermöglicht zudem eine Verwaltung und Statusübersicht über alle generierten Rechnungen.

### 2.2.5 Projektbeteiligte

In diesem Unterkapitel sind die internen Projektbeteiligten aufgeführt. Jedem Teammitglied werden mehrere Rollen zugewiesen, die er im Laufe des Projekts wahrzunehmen hat

Projektleitung -> Markus Kaufmann

Entwickler -> Nicolas Hässig

Analysten -> Fabrizio Piacente, Nicolas Hässig

Architekt -> Markus Kaufmann

Tester/ Qualitätskontrolle -> Fabrizio Piacente, Nicolas Hässig und Markus Kaufmann

### 2.2.6 Projektplanung: 1. Iteration

Unter diesem Punkt werden die Ziele und Anforderungen an das Projekt formuliert. Dabei unterscheiden wir bewusst zwischen Projektanforderungen (interne Ziele) und funktionale Ziele an das Projekt.

Anforderungen an das Projekt:

* Festlegung der Projektrichtlinien
* interner Aufbau und Verwaltung
* klare Abläufe sind definiert
* Meilensteine sind definiert (Zeitschiene)

Ziele des Projektes:

* Your SwissQR kann den Anforderungen (Six) gemäss erstellt werden
* Es wird eine übersichtliche Verwaltungssoftware mit ausgeliefert (GUI für die Übersicht und das Management)
* KMU's und Einzelunternehmer sollen auch profitieren
* Einfache Anbindung an bestehende Systeme
* Unabhängige Funktionalität (Ohne Drittsysteme)

## Anforderungsanalyse

In diesem Kapitel wird die Anforderungsanalyse für die SwissQR Software der Koch AG durchgeführt und näher beschrieben.

### 2.3.1 Vision

Da ab 2022 unabdinglich wird, den Swiss QR einzusetzen braucht es eine Branchenunabhängige Lösung für alle Rechnungssteller.

Your SwissQR bietet eine Möglichkeit für jeden Rechnungssteller eine Rechnung an seine Kunden auszuliefern. Dies mit geringen Kosten und geringem administrativ Aufwand.

Mit diesem Produkt will die Koch AG Branchenführer für das Erzeugen und Verwalten von SwissQR Rechnungen sein. Die Vision lautet deshalb:

**SwissQR mit Koch AG**

### 2.3.2 Name

Da der Name mitentscheidend für den Erfolg eines Produkts ist, wurde ein Name gewählt, der sowohl kurz und prägnant ist, aber auch beschreibt, für was die Software genutzt werden kann. Der Name lautet deshalb:

**YourSwissQR**

Ein entsprechendes Logo ist noch zu designen

### 2.3.3 Stakeholder

Im Kapitel Stakeholder werden sämtlich internen und externen Personen und Gruppen genannt, die in irgendeiner Form am Projekt beteiligt sind.

* Projektbeteiligte (Siehe Kapitel 2.2.5)
  + Sind für die Planung (unter Berücksichtigung der Interessen der weiteren Stakeholder), die Entwicklung, die Auslieferung und Wartung des Projektes verantwortlich
* Auftraggeber; BistroTreuhand AG
  + Ist der Auftraggeber, resp. der Endkunde des Projekts, dessen Ansprüche bei der Entwicklung berücksichtigt werden müssen.
* Six Group
  + Der SwissQR Code muss gemäss den Richtlinen der Six Finance Group erzeugt werden, zudem stehen sie bei allfälligen Fragen zur Verfügung und stellen auch Validierungsportale für den SwissQR zur Verfügung
* Kantonale Behörden
  + Die Rechnungsstellung, resp. deren rechtliche Grundlage ist Sache der Kantonalen Behörden, weshalb deren Vorgaben mit Berücksichtigt werden müssen.

### 2.3.4 Funktionale Anforderungen

In diesem Abschnitt werden die Use Cases der YourSwissQR Applikation aufgelistet.

|  |  |
| --- | --- |
| ID: | 1 |
| Priorität | 1 |
| Titel: | Rechnung erstellen |
| Beschreibung: | Der Benutzer erstellt über einen Button/Shortcut eine neue Rechnung, die Eingabemaske öffnet sich. |
| Anwender: | Benutzer |
| Voraussetzung: | Benutzerrecht «Neu erstellen» |
| Erfolgsfall: | Neue Rechnung wird mit den angegebenen Informationen erzeugt und in der Oberfläche dargestellt. Der SwissQR zur Rechnung wird generiert. |

Tabelle Use Case Rechnung erstellen

|  |  |
| --- | --- |
| ID: | 2 |
| Priorität | 1 |
| Titel: | Rechnung bearbeiten |
| Beschreibung: | Der Benutzer selektiert eine bestehende Rechnung und öffnet die Bearbeitungsmaske über Button/Shortcut |
| Anwender: | Benutzer |
| Voraussetzung: | Selektierte Rechnung, Benutzerrecht «Bearbeiten» |
| Erfolgsfall: | Maske öffnet sich, Änderungen werden nach Bestätigung übernommen. Der SwissQR Code der Rechnung wird neu erstellt. |

Tabelle Use Case Rechnung bearbeiten

|  |  |
| --- | --- |
| ID: | 3 |
| Priorität | 1 |
| Titel: | Rechnung löschen |
| Beschreibung: | Der Benutzer selektiert eine bestehende Rechnung und kann diese über Button/Shortcut löschen |
| Anwender: | Benutzer |
| Voraussetzung: | Selektierte Rechnung, Benutzerrecht «Löschen» |
| Erfolgsfall: | Rechnung ist gelöscht und wird nicht mehr angezeigt. |

Tabelle Use Case Rechnung löschen

|  |  |
| --- | --- |
| ID: | 4 |
| Priorität | 2 |
| Titel: | Rechnungsstatus setzen |
| Beschreibung: | Der Benutzer kann auf der Bearbeitungsmaske den Status der Rechnung setzen (Eröffnet, verschickt u.ä.) |
| Anwender: | Benutzer |
| Voraussetzung: | Selektierte Rechnung, zur Bearbeitung geöffnet, Benutzerrecht «Bearbeiten» |
| Erfolgsfall: | Status wird korrekt angewandt und dargestellt. |

Tabelle 10 Use Case Status ändern

|  |  |
| --- | --- |
| ID: | 5 |
| Priorität | 3 |
| Titel: | Rechnung als PDF erstellen |
| Beschreibung: | Der Benutzer kann über einen Button/Shortcut die Rechnung als PDF generieren. |
| Anwender: | Benutzer |
| Voraussetzung: | Selektierte Rechnung |
| Erfolgsfall: | PDF wird korrekt (siehe Styleguide Six Finance) erzeugt |

Tabelle Use Case PDF erstellen

|  |  |
| --- | --- |
| ID: | 6 |
| Priorität | 3 |
| Titel: | Rechnung drucken |
| Beschreibung: | Der Benutzer kann über einen Button/Shortcut die Rechnung drucken, wobei der Standard Drucker des Systems verwendet wird |
| Anwender: | Benutzer |
| Voraussetzung: | Selektierte Rechnung |
| Erfolgsfall: | Bericht wird korrekt gedruckt (siehe Styleguide Six Finance) |

Tabelle Use Case Drucken

|  |  |
| --- | --- |
| ID: | 7 |
| Priorität | 4 |
| Titel: | Rechnung versenden |
| Beschreibung: | Der Benutzer kann über einen Button/Shortcut die Rechnung als PDF versenden. |
| Anwender: | Benutzer |
| Voraussetzung: | Selektierte Rechnung |
| Erfolgsfall: | Mailclient des Systems öffnet sich, Datei befindet sich als PDF im anhang |

Tabelle Use Case Versenden

### 2.3.5 Nicht-funktionale Anforderungen

Die nicht funktionalen Anforderungen an die YourSwissQR Applikationen werden hier aufgeführt.

* Entspricht den Richtlinien von Six Group
* Entspricht den gesetzlichen Vorgaben
* Kostengünstiges Produkt
* Entspricht den UX Anforderungen (einfach logisch klar)
* Ist ausbaubar
* Anbindung an andere Module möglich

### 2.3.6 Benutzer

Unter dem Kapitel Benutzer sind sämtliche Personen aufgelistet und beschrieben, die die YourSwissQR Applikation in irgendeiner Form nutzen werden.

|  |  |
| --- | --- |
| Bezeichnung | Beschreibung |
| Mitarbeiter der Buchhaltung | Da das Debitorenwesen primär Sache der Buchhaltung der jeweiligen Firma ist, stellt die Gruppe der Buchhalter die Hauptverwendungsgruppe dar. |
| Sachbearbeiter | Sachbearbeiter, welche selbständig Rechnungen stellen dürfen sind die zweite Anwendergruppe von YourSwissQR |
| Projektarbeiter der KochAG | Projektleiter der KochAG werden vor allem Konfigurationsaufgaben und Support der YourSwissQR Applikation vornehmen, weshalb sie hier ebenfalls aufgelistet werden. |

Tabelle Benutzer von YourSwissQR

### 2.3.7 Umsysteme

In diesem Unterkapitel sind sämtliche Umsysteme der YourSwissQR Applikationen aufgelistet. Als Umsysteme sind hier insbesondere Anwendungen gemeint, die eine Schnittstelle zur YourSwissQR Applikation aufweisen, oder sonstige Abhängigkeiten zu dieser haben.

* Verwendetes ERP (z.B Auftragsabwicklungssystem, wenn in Verwendung)
  + Falls sonstige ERP Systeme im Einsatz sind, kann/sollte die Rechnungsstellung mit der YourSwissQR Applikation synchronisiert werden.
* Buchhaltungssoftware
  + Falls Buchhaltungssoftware im Einsatz ist, ist eine Importmöglichkeit der Rechnungen aus der Koch AG Software von Vorteil.
* Datenbank
  + Um Rechnungsinformationen langfristig persistent zu speichern, muss eine Datenbankanbindung realisiert werden
* Steuersystem
  + Analog Buchhaltungssoftware
* Mailclient
  + Das Versenden von Rechnungen (siehe Use Cases) bedingt einen funktionierenden Mailclient.
* Drucker
  + Das Drucken von Rechnungen (siehe Use Cases) bedingt einen funktionierenden und konfigurierten Drucker.

### 2.3.8 Systemkontextdiagramm

Das Systemkontextdiagramm, welches aus den Umsystemen, den Benutzern und den Usecases definiert wurde, ist untenstehend als Grafik ersichtlich

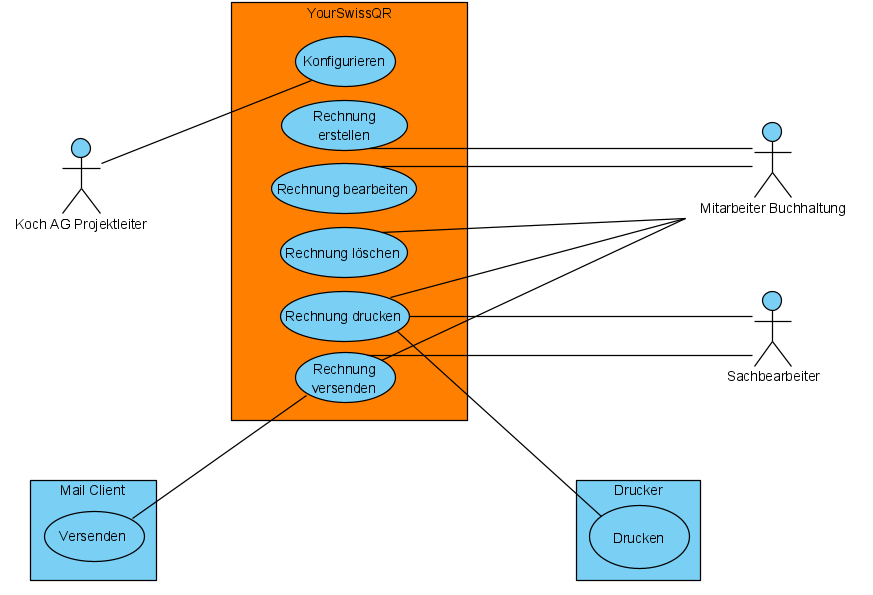


Abbildung Systemkontext Diagramm

# Aufgabe 3: Objektorientierte Analyse (OOA)

Der SwissQR Code wird in einer Objekt Orientierten Umgebung umgesetzt. Dadurch werden Daten und Programmcode in übersichtlichen, wiederverwendbaren Modulen, sprich Objekten gekapselt. Dies reduziert den Programmieraufwand und hilft, Fehler zu vermeiden.

## Fachklassenmodell

SwissQR beinhaltet insgesamt 8 Klassen, die mit entsprechenden Variablen befüllt sind. Die Anzahl an Klassen ist schlank gehalten, damit die Übersicht gewährleistet ist und es einfach zu Lesen bleibt.

Unten ersichtlich als UML – Fachklassenmodell:

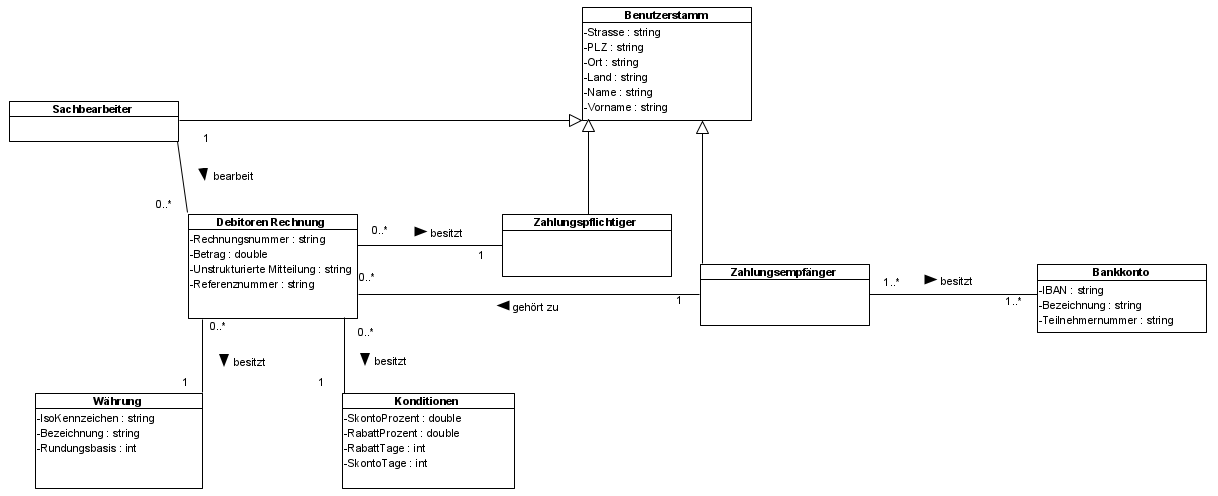


Abbildung Fachklassendiagramm

## Zustandsmodell

In diesem Kapitel die Zustände der beiden ausgesuchten UseCases (Rechnung erstellen und Rechnung bearbeiten) dargestellt.

Dieses wurde als UML-Zustandsmodell realisiert.

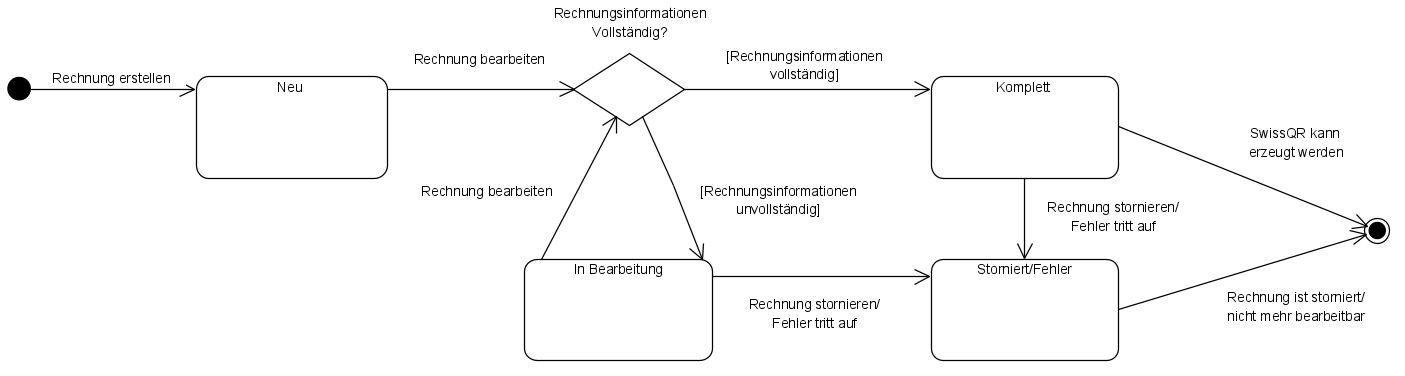


Abbildung Zustandsmodell

## Verantwortlichkeit Fachklassen

Im Folgenden Abschnitt werden die verschiedenen Verantwortlichkeiten der Klassen geregelt. Mit einem kleinen Kurzbeschrieb der Verwantwortlichkeit pro Klasse.

|  |  |
| --- | --- |
| Fachklasse | Beschreibung |
| Sachbearbeiter | Person die die Rechnung bearbeitet |
| Währung | Enthält Informationen zur Währung |
| Debitoren Rechnung | Basisklasse für die Erstellung des SwissQR Codes |
| Konditionen | Enthält Informationen zu Vergünstigungen und Sonderpreisen |
| Zahlungspflichtiger | Person/Firma/Organisation, die die Rechnung bezahlt |
| Zahlungsempfänger | Person/Firma/Organisation, die die Rechnung ausstellt, resp. den Zahlungsbetrag erhält. |
| Benutzerstamm | Klasse für Daten von Person/Firma/Organisation |
| Bankkonto | Enthält kontorelevante Informationen wie IBAN |

Tabelle Fachklassen

# Aufgabe 4: Designmodell erstellen (OOD)

In dieser Aufgabe muss für das SWISSQR Projekt ein Designmodell erarbeitet werden. Dies wird in Visual Paradigm umgesetzt.

Dazu gibt es teilweise zu einer Schicht eine erste und zweite Iteration. Entsprechend wie sich das Projekt weiterentwickelt hat.

## Designmodell für die Persistence Schicht

Das Designmodell wird als wesentlicher Input für Aktivitäten bei der Implementierung und beim Testen verwendet. UML-Darstellung: Modell, stereotypisiert als «Designmodell».

### Erste Iteration des Designmodell für die Persistence Schicht

In der ersten Iteration wurde der grundsätzliche Aufbau der Persistenz-Schicht ermittelt, ohne die später zu implementierenden Methoden im Detail auszuarbeiten

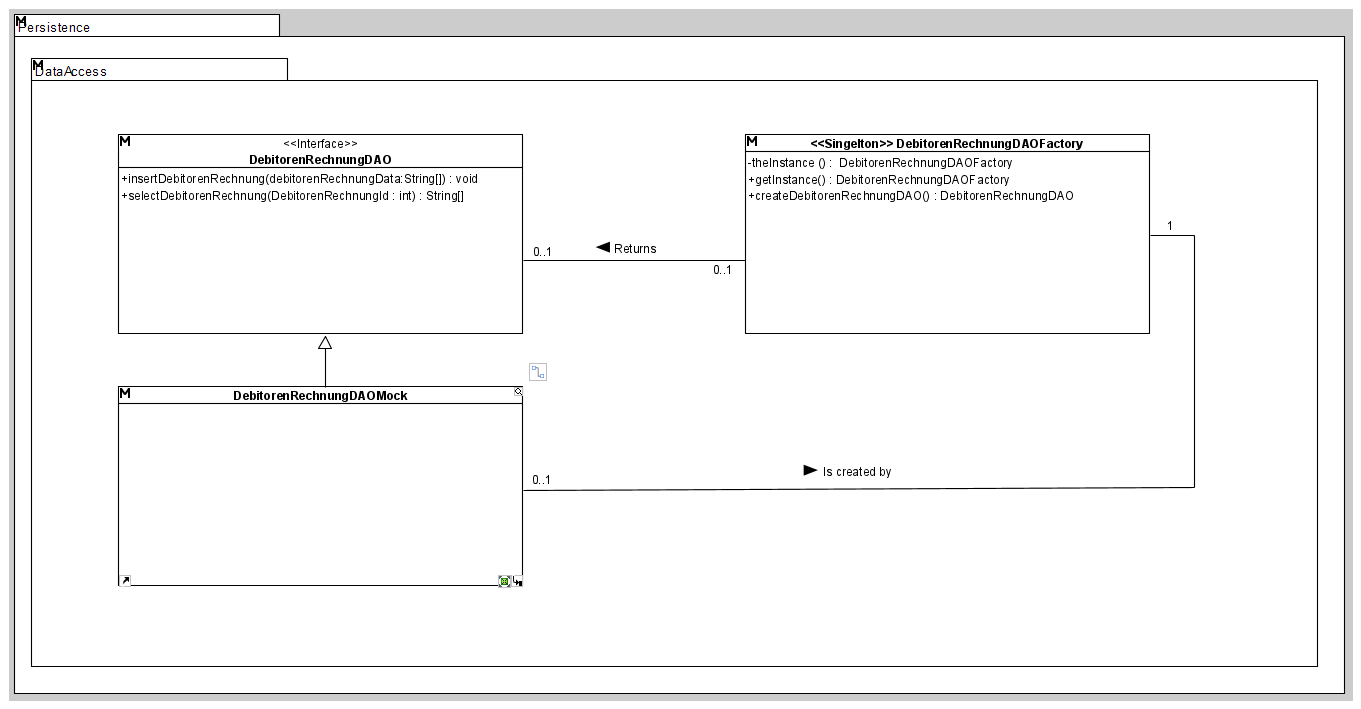
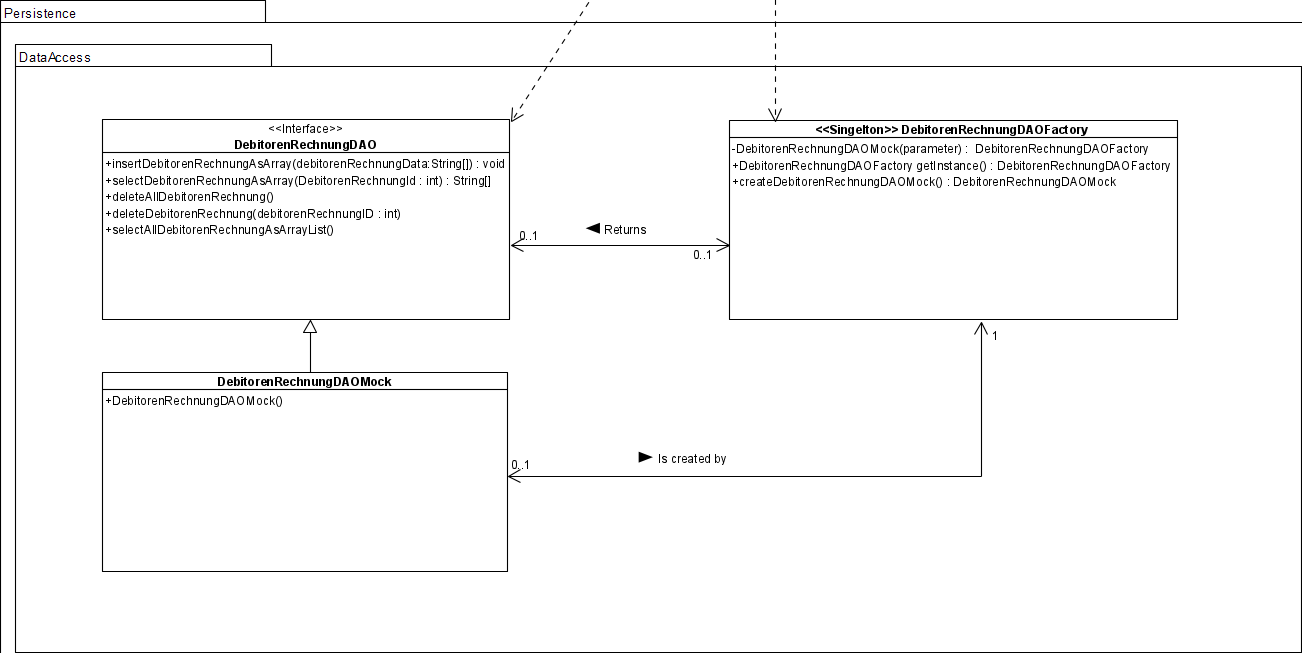


Abbildung Designmodell für die Persistence Schicht

### Zweite Iteration des Designmodell für die Persistence Schicht

Die zweite Iteration zeigt die tatsächliche Umsetzung der Persistenz Schicht mit allen Methoden und Parametern.



## Designmodell für die Business-Schicht

### Erste Iteration des Designmodell für die Business-Schicht

Die erste Iteration der Business Schicht zeigt das Komplette Fachklassenmodell der geplanten Umsetzung ohne Methoden im Detail auszuarbeiten.

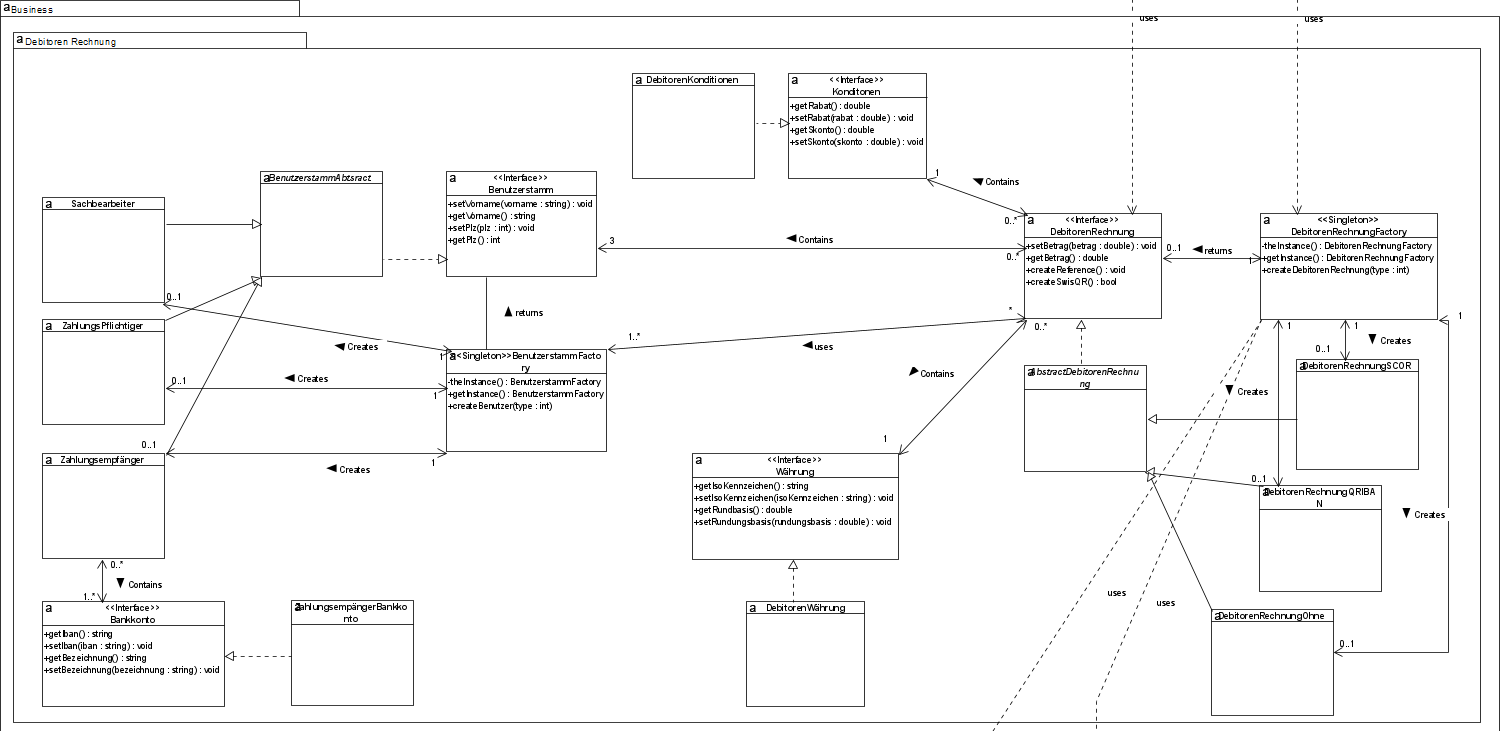
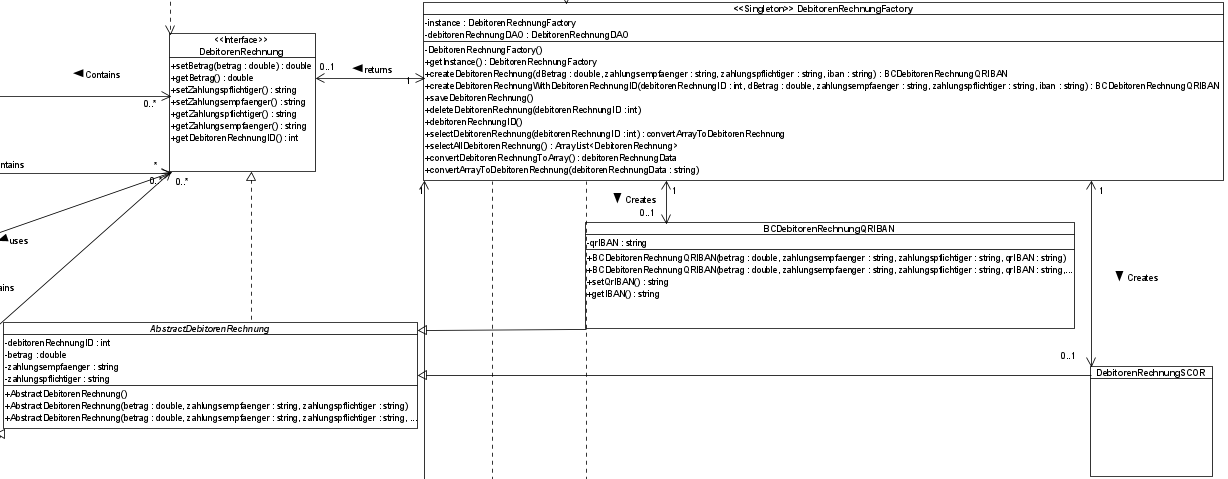


Abbildung Designmodell für die Business-Schicht

### Zweite Iteration des Designmodell für die Business-Schicht

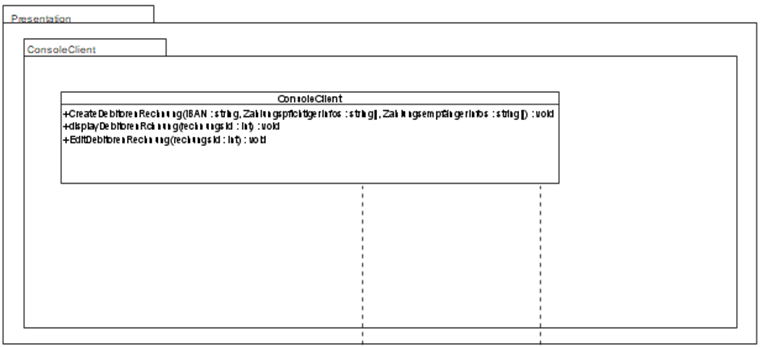
Die zweite Iteration der Businessschicht zeigt die tatsächliche Umsetzung. Diese beinhaltet lediglich sämtlich Klassen zur Debitorenrechnung selbst, weitere Klassen wir zum Beispiel der Zahlungsempfänger sind als «Simple Type Member» in der Debitorenrechnung selbst enthalten. Dafür sind die detaillierten Informationen zu den Klassen selbst enthalten.



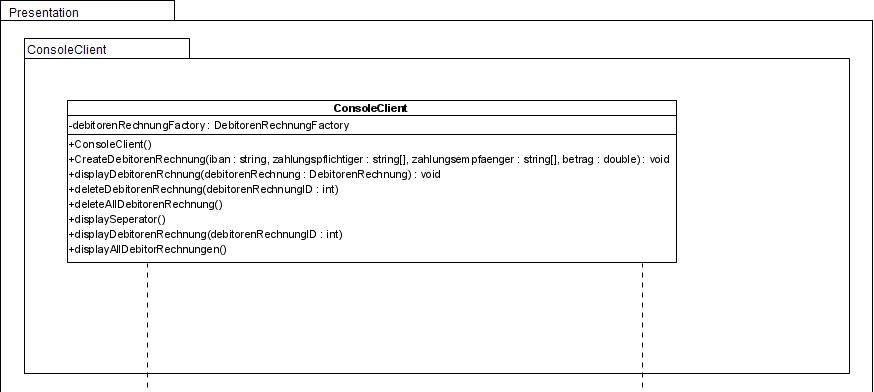
## Designmodell für die Presentation Sicht

Für diese Applikation wird kein komplexes GUI verwendet, sondern lediglich eine Konsolenausgabe, wie sie in der untenstehenden Abbildung des Presentation-Layers ersichtlich ist.

### Erste Iteration des Designmodells für die Presentation Sicht



### Erste Iteration des Designmodells für die Presentation Sicht



## Integriertes Designmodell

Das Integrierte Designmodell zeigt, wie die einzelnen Schichten zusammenhängen und miteinander verknüpft sind. Es wird hier auf das klassische drei Schichtenmodell, Presentation, Logic, Persistence zurückgegriffen.

Das integrierte Designmodell bezieht sich jeweils nur auf die erste Iteration. Die zweite Iteration ist in Kapitel 4.1 bis 4.3 jeweils mit aufgeführt.

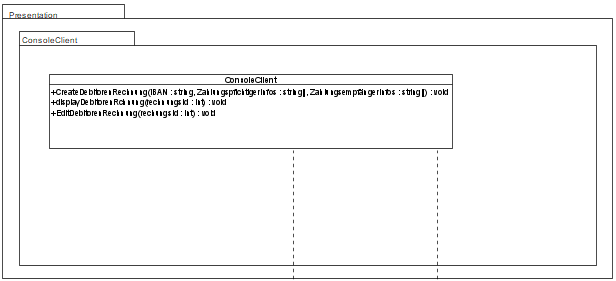


Abbildung Integriertes Designmodell 1/3

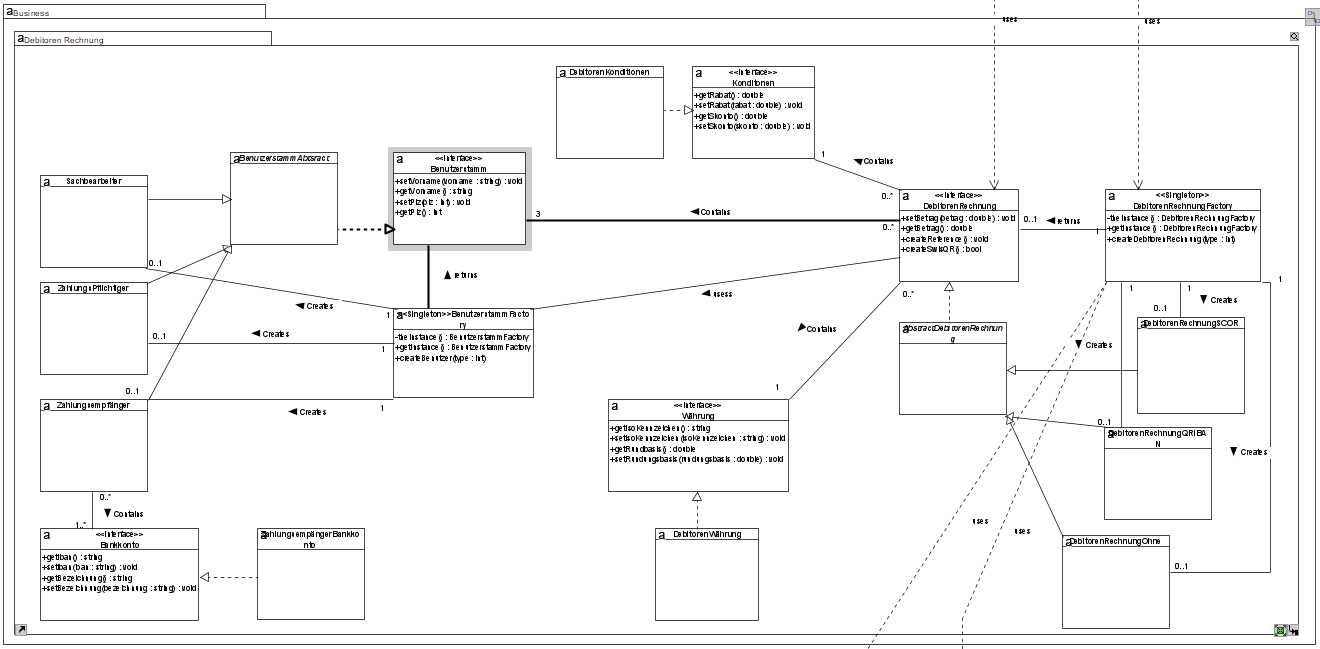


Abbildung Integriertes Designmodell 2/3

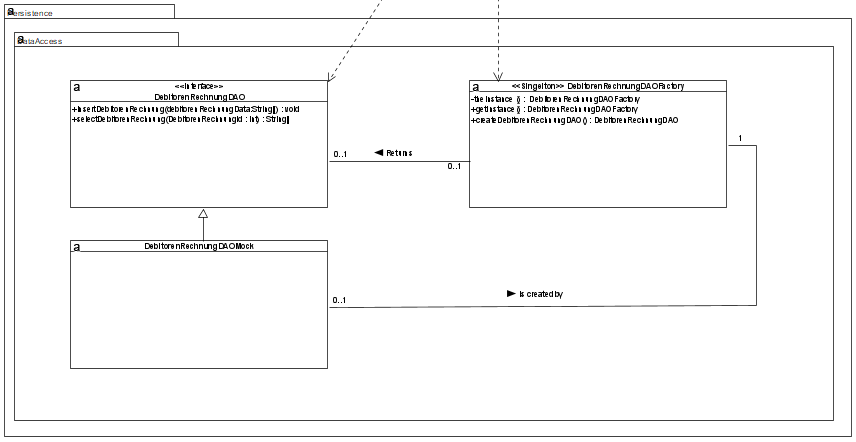


Abbildung Integriertes Designmodell 3/3

## Sequenz Diagramm

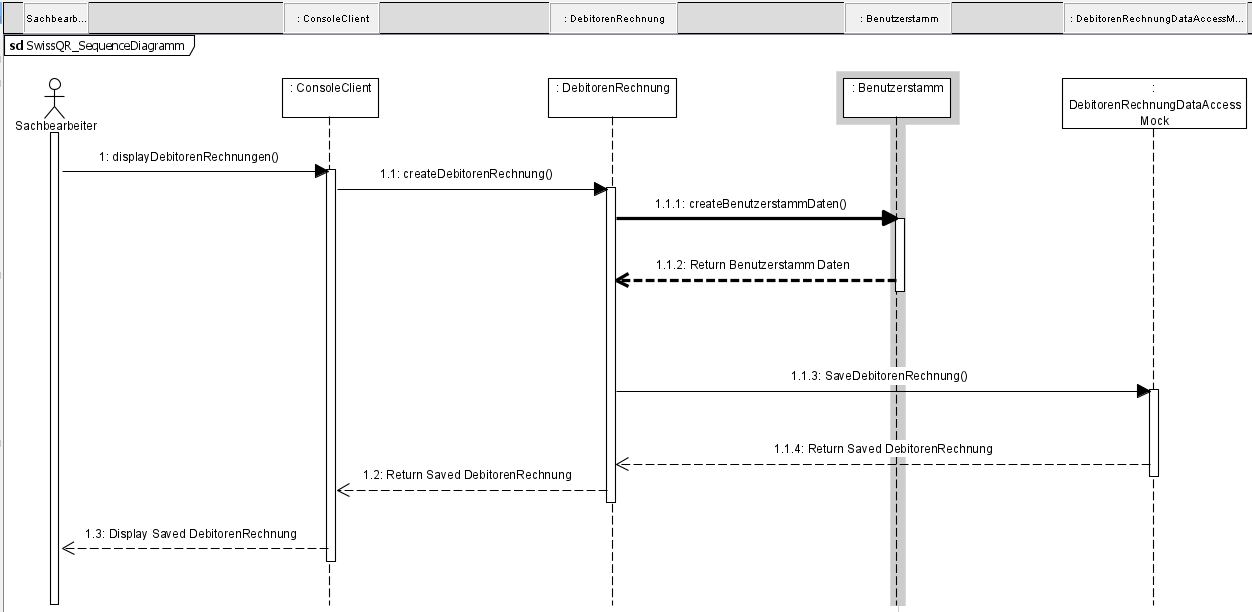


Abbildung Sequenz Diagramm

## Komponenten Diagramm

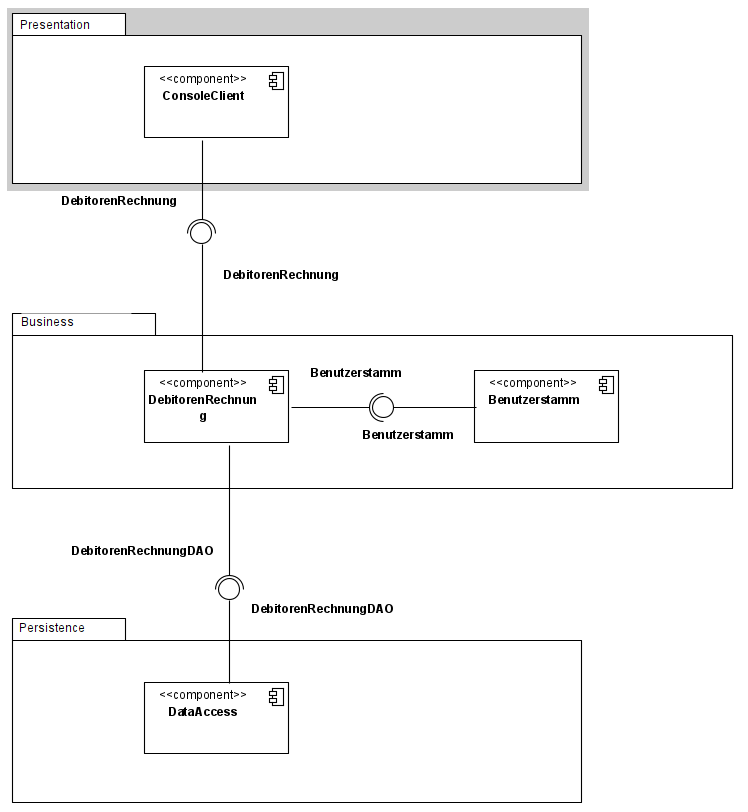


Abbildung Komponenten Diagramm

# Glossar

## Begriffserklärung

|  |  |
| --- | --- |
| Begriff | Definition/Erklärung |
| Designmodell | Fachklassenmodell angereichert um technische Aspekte. Bezieht  sich primär auf nicht-funktionale Anforderungen. |
| Eclipse | Integrierte Entwicklungsumgebung (Open Source) für diverse Programmiersprachen, im Rahmen dieses Projektes als Programmierwerkzeug für JAVA verwendet |
| ERP | Enterprise-Resource-Planning (ERP) bezeichnet die unternehmerische Aufgabe, Ressourcen wie Kapital, Personal, Betriebsmittel, Material und Informations- und Kommunikationstechnik im Sinne des Unternehmenszwecks rechtzeitig und bedarfsgerecht zu planen, steuern und verwalten. |
| Fachklassenmodell | Zeigt wichtigste Gegenstände (Entities, Geschäftsobjekte) einer  Domäne und deren Beziehungen. Bezieht sich primär auf  funktionale Anforderungen. |
| Git | Git ist ein verteiltes Versionierungssystem welches frei als Open-Source zur Verfügung gestellt wird. |
| GitHub | GitHub ist ein netzbasierter Dienst zur Versionsverwaltung für Software-Entwicklungsprojekte. |
| GUI | GUI steht für "Graphical User Interface". Man versteht darunter die grafische Oberfläche, die Sie nutzen, um ein Spiel bzw. Programm zu bedienen. |
| IBAN | Die Internationale Bankkontonummer ist eine internationale, standardisierte Notation für Kontonummern. |
| Java | Java ist eine objektorientierte Programmiersprache, die für den Einsatz in der verteilten Umgebung des Internets entwickelt wurde. Mit Java können komplette Anwendungen erstellt werden, die auf einem einzelnen Computer oder verteilt zwischen Servern und Clients in einem Netzwerk ausgeführt werden. Die Sprache kann auch genutzt werden, um kleine Anwendungsmodule oder Applets als Teil einer Webseite zu bauen. |
| Klasse | Die Klasse repräsentiert einen Bauplan, der als Grundlage der objektorientierten Programmierung dient. Sie liefert die Definition dessen, was ein bezeichnetes Objekt innerhalb dieses Programmierstils ausmacht. |
| Klassendiagramm | Zeigt Klassen und ihre Beziehungen. |
| Komponente | Ein kontextfrei lauffähiges Stück Software mit definierter Schnittstelle |
| Komponenten Diagramm | Zeigt Komponenten und ihre Beziehungen. |
| Mailclient | Als Mail-Client bezeichnet man ein Programm, mit dem E-Mails empfangen, gelesen, geschrieben und versendet werden |
| Maske | Maske bzw. Bildschirmmaske entsprechen Formularen, 131§sauf den bestimmten Informationen bereits enthalten sind, während andere noch eingetragen werden müssen. |
| Notepad++ | Texteditor für Windows, welcher die Syntax diverser Programmiersprachen unterstützt. |
| SIX | SIX (Swiss Infrastructure and Exchange) betreibt die Infrastruktur für den Finanzplatz Schweiz. Das Unternehmen erbringt Dienstleistungen rund um Wertpapiergeschäfte, die Aufbereitung von Finanzinformationen, den Zahlungsverkehr und baut eine digitale Infrastruktur auf. |
| use case | Ein Anwendungsfall bündelt alle möglichen Szenarien, die eintreten können, wenn ein Akteur versucht, mit Hilfe des betrachteten Systems ein bestimmtes fachliches Ziel zu erreichen. Er beschreibt, was inhaltlich beim Versuch der Zielerreichung passieren kann und abstrahiert von konkreten technischen Lösungen. |
| UML | Die Unified Modeling Language (UML) bezeichnet eine visuelle Modellierungssprache, die sich über die Bereiche Architektur, Design und die Abläufe von Herstellungsprozessen erstreckt. Konzeptionell handelt es sich um eine grafische Standard-Notation, das heißt, ein spezielles Aufzeichnungssystem mit eigenem Vorrat an Zeichen und Symbolen. |
| UX | Mit dem Begriff User Experience (deutsch: Nutzererlebnis, kurz: UX) beschreibt man das Erlebnis, das der Anwender bei der Nutzung eines Produktes hat. |
| Repository | Der Begriff Repository steht allgemein für Aufbewahrungsort oder Behälter. In der Softwaretechnik handelt es sich um eine zentrale Ablage in dem digitalen Daten, Dokumente, Objekte und Programme mit ihren Metadaten verwaltet werden. |
| Sequenzdiagramm | Zeigt die Zusammenarbeit von Objekten innerhalb eines Anwendungsfalls (Use Case). |
| Smartgit | Windows-Client für den Zugriff auf Git Repositories. |
| SwissQR | Die QR-Rechnung ist ein Standard der Schweizer Finanzindustrie für maschinenlesbare Rechnungen. Sie ersetzt seit dem 30. Juni 2020 schrittweise die verschiedenen bisher benutzten ESR/VESR Einzahlungsscheine. |
| Visual Paradigm | Visual Paradigm ist eine UML-Software welche die Modellierung von Software mit weit verbreiteten Modellierungssprachen wie UML und SysM ermöglicht. |
| Zustandsmodell | Zeigt die Zustände eines Objekts über verschiedene Anwendungsfälle (Use Cases) hinweg |

## Klassenübersicht

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Model ID | Layer | Name | Stereotypes | Visibility | Description | Abstract | Parent ID |
| 3 | Presentation | ConsoleClient |  | public |  | No | 4 |
| 6 | Persistence | DebitorenRechnungDAO | <<Interface>> | public |  | No | 7 |
| 9 | Persistence | DebitorenRechnungDAOFactory |  | public |  | No | 7 |
| 11 | Persistence | DebitorenRechnungDAOMock |  | public |  | No | 7 |
| 13 | Business | Zahlungsempfänger |  | public |  | No | 14 |
| 16 | Business | ZahlungsPflichtiger |  | public |  | No | 14 |
| 18 | Business | Sachbearbeiter |  | public |  | No | 14 |
| 20 | Business | BenutzerstammAbtsract |  | public |  | Yes | 14 |
| 22 | Business | BenutzerstammFactory |  | public |  | No | 14 |
| 24 | Business | Benutzerstamm | <<Interface>> | public |  | No | 14 |
| 26 | Business | ZahlungsempängerBankkonto |  | public |  | No | 14 |
| 28 | Business | DebitorenKonditionen |  | public |  | No | 14 |
| 30 | Business | DebitorenWährung |  | public |  | No | 14 |
| 32 | Business | Währung | <<Interface>> | public |  | No | 14 |
| 34 | Business | Bankkonto | <<Interface>> | public |  | No | 14 |
| 36 | Business | Konditonen | <<Interface>> | public |  | No | 14 |
| 38 | Business | DebitorenRechnungOhne |  | public |  | No | 14 |
| 40 | Business | DebitorenRechnungSCOR |  | public |  | No | 14 |
| 42 | Business | DebitorenRechnungQRIBAN |  | public |  | No | 14 |
| 44 | Business | AbstractDebitorenRechnung |  | public |  | Yes | 14 |
| 46 | Business | DebitorenRechnungFactory |  | public |  | No | 14 |
| 48 | Business | DebitorenRechnung | <<Interface>> | public |  | No | 14 |

# Tabellen und Abbildungsverzeichnis

## Abbildungsverzeichnis

[Abbildung 1 Systemkontext Diagramm 17](#_Toc56976992)

[Abbildung 2 Fachklassendiagramm 18](#_Toc56976993)

[Abbildung 3 Zustandsmodell 19](#_Toc56976994)

[Abbildung 4 Designmodell für die Persistence Schicht 20](#_Toc56976995)

[Abbildung 5 Designmodell für die Business-Schicht 20](#_Toc56976996)

[Abbildung 6 Integriertes Designmodell 1/3 21](file:///C:\Users\fabri\Documents\GitHub\swiss-qr-code\Documentation\ProjectWork.docx#_Toc56976997)

[Abbildung 7 Integriertes Designmodell 2/3 22](file:///C:\Users\fabri\Documents\GitHub\swiss-qr-code\Documentation\ProjectWork.docx#_Toc56976998)

[Abbildung 8 Integriertes Designmodell 3/3 23](file:///C:\Users\fabri\Documents\GitHub\swiss-qr-code\Documentation\ProjectWork.docx#_Toc56976999)

[Abbildung 9 Sequenz Diagramm 24](file:///C:\Users\fabri\Documents\GitHub\swiss-qr-code\Documentation\ProjectWork.docx#_Toc56977000)

[Abbildung 10 Komponenten Diagramm 25](#_Toc56977001)

## Tabellenverzeichnis:

[Tabelle 1 Richtlinien und Schreibweisen 5](#_Toc56977002)

[Tabelle 2 Eingesetzte Entwicklungssoftware 7](#_Toc56977003)

[Tabelle 3 Weitere verwendete Software 7](#_Toc56977004)

[Tabelle 4 Checkliste und Installationsanleitung Eclipse 7](#_Toc56977005)

[Tabelle 5 Projektentscheidungen 9](#_Toc56977006)

[Tabelle 6 Koch AG Übersicht 10](#_Toc56977007)

[Tabelle 7 Use Case Rechnung erstellen 14](#_Toc56977008)

[Tabelle 8 Use Case Rechnung bearbeiten 14](#_Toc56977009)

[Tabelle 9 Use Case Rechnung löschen 14](#_Toc56977010)

[Tabelle 10 Use Case Status ändern 15](#_Toc56977011)

[Tabelle 11 Use Case PDF erstellen 15](#_Toc56977012)

[Tabelle 12Use Case Drucken 15](#_Toc56977013)

[Tabelle 13 Use Case Versenden 15](#_Toc56977014)

[Tabelle 14 Benutzer von YourSwissQR 16](#_Toc56977015)

[Tabelle 15 Fachklassen 19](#_Toc56977016)

# Quellenverzeichnis

Kursunterlagen Software Engineering 1 (Juventus)

Analyse und design mit UML 2.5

Oesterreich, Bernd (2013): Analyse und design mit UML 2.5 - Objektorientierte Softwareentwicklung; 11. Auflage; Oldenburg Verlag

Codingstyle [31.09.2020]

<https://google.github.io/styleguide/javaguide.html>

JDK 14 Documentation [02.01.2021]

https://docs.oracle.com/en/java/javase/14/

Style Guide QR-Rechnung [02.01.2021]

<https://www.paymentstandards.ch/dam/downloads/style-guide-de.pdf>

Swiss Payment Standards 2019 [02.01.2021]

<https://www.paymentstandards.ch/dam/downloads/change-documentation-qrr-de.pdf>